Oracle® Cloud

Migrating Your Web and Mobile Applications to Oracle Visual Builder Studio

F21007-05
September 2025

Oracle Cloud Migrating Your Web and Mobile Applications to Oracle Visual Builder Studio,

F21007-05

Primary Author: Oracle Corporation

Contents

About This Content

This guide describes how to migrate visual applications from Oracle Visual Builder instances to Oracle Visual Builder Studio.

Audience

Migrating Your Web and Mobile Applications to Visual Builder Studio is intended for users who need to migrate visual applications from Oracle Visual Builder instances to Oracle Visual Builder Studio.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Resources

See these Oracle resources:

- Oracle Public Cloud

  `http://cloud.oracle.com`
- About Oracle Visual Builder in *Developing Applications with Oracle Visual Builder*
- About Oracle Cloud in *Getting Started with Oracle Cloud*

Conventions

The following text conventions are used in this document.

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Learn About Migrating to Oracle Visual Builder Studio

Learn about the benefits of developing your web and mobile applications using VB Studio, and understand the process of migrating your applications from Oracle Visual Builder to VB Studio.

Why Migrate to Oracle Visual Builder Studio

Oracle Visual Builder Studio (VB Studio) combines Oracle Visual Builder's visual and declarative cloud environment for developing mobile and web applications with the developer and lifecycle management tools of Developer Cloud Service.

VB Studio provides tools supporting all the stages of the app dev lifecycle: design, build, test, and deploy. This includes integrated Git repositories, issue tracking and code review tools and an integrated build system for continuous development and deployment of your apps.

With VB Studio you get:

- Built-in repositories for hosting code in Git and for hosting binaries, such as Maven dependencies
- The full Oracle Visual Builder Designer, that is now tightly integrated with your Git repositories, so that developers can manage changes, apply version control best practices, and collaborate with their teammates to develop applications
- A continuous integration service so you can automate your build and test systems
- A continuous delivery service that tightly integrates with Oracle Cloud Applications
- The ability to build and display different flavors of the UI to meet the needs of discrete users of certain Oracle Cloud Applications (those built with VB Studio and Oracle Java Extension Toolkit (Oracle JET)), also within a Git framework
- Agile boards and an issue tracking system for tracking sprints, tasks, defects, and features

VB Studio enables developers to easily deploy their applications to their preferred target, whether it’s a staging or production instance of Oracle Cloud Applications or an Oracle Cloud Infrastructure (OCI) service instance.

About Working in Visual Applications in VB Studio

The way projects and your visual applications are organized in VB Studio is slightly different. Here are some key differences of the VB Studio ecosystem that you'll need to get familiar with:

- Within a single VB Studio instance, you and your team members who use that instance are considered an *organization*. Within your organization, you will likely belong to one or more *projects*, each of which is devoted to a discrete software effort. For example, you might have a project for building a new Financial mobile app, and a different project for creating an HR web application. A project brings together all the tools you need to create those artifacts, such as a Git repository for storing your source code, a pipeline to provide continuous integration and delivery, an issue tracking system, team wikis, and more.

- All of your work in VB Studio is done in the context of a *workspace*, a completely private area where you can work on your visual application. Your work is stored in your own clone of the project's Git repository, and is not visible to others until you a) merge it to the project's Git repo, b) choose to Share it with other for testing, or c) deploy it. A workspace also includes a pointer to the development or test environment where you plan to deploy your app, which must be a separate Visual Builder instance (also known as a "runtime environment"). You can deploy your app manually, or wire it up to a pipeline to do it automatically, such as when a developer on your project merges his or her branch to the master branch.

While the Visual Builder's Designer is still available, you can see that some things have changed. For details about tasks that you now do differently in VB Studio, see For Visual Builder Users.

About the Migration Scope

You can migrate the sources and data of your visual application either as an archive that you export from Oracle Visual Builder or by cloning the existing repository where your visual application sources and data are stored.

If you export an archive, the archive file contains the design-time metadata for the applications in your visual application, plus a variety of other files that your visual application needs once it is imported in the new instance. The following directory structure for a visual application that contains a web and mobile app, plus a business object provides an illustrative example of the type of metadata that an archive file contains.

```
VisualApplicationArchiveDirectory
+---businessObjects
|   \---Department
+---mobileApps
|   \---hrmobileapp
|       +---flows
|       |   \---main
|       |       \---pages
|       +---pages
|       |   \---resources
|       |       \---strings
|       |           \---app
|       |               \---nls
|       |                   \---root
|       ...
|       \---settings
|           \---mobile-build-templates
+---process
|   \---pcs
+---services
+---settings
\---webApps
    \---hrwebapp
        +---flows
        |   \---main
        |       \---pages
        +---pages
        |   \---resources
        ...
        +---resources
        |   +---css
        |   \---strings
        |       \---app
        |           \---nls
        |               \---root
        \---settings
Gruntfile.js
package.json
visual-application.json
```

When you export the visual application you can choose if you want the archive to include the development data contained in the application's custom business objects. Some information, such as the user credentials to access external REST endpoints and artifacts that are part of the mobile configuration (such as keystore, iOS provisioning profiles, and passwords) are not included when you export a visual application, so you will need to re-enter these after you import the application.

Apart from the design-time metadata for each visual application, you also need to export the application data for live applications. Before you migrate this data, you need to lock the live application which prevents end users from accessing the application.

2

Prepare to Migrate Oracle Visual Builder Visual Applications to Oracle Visual Builder Studio

Before you migrate Oracle Visual Builder visual applications to Oracle Visual Builder Studio, plan and prepare for migration.

About Downtime Requirements

For most of the migration process, the availability of your existing Oracle Visual Builder applications is not affected.

Your Oracle Visual Builder instance continues to run and can serve client requests while you complete the pre-migration tasks in Oracle Visual Builder Studio. Some downtime is required after you complete the pre-migration tasks as you need to transfer application data from the Oracle Visual Builder instance to Oracle Visual Builder Studio. Before you attempt to transfer the application data, lock the live application on the Oracle Visual Builder instance so that application users do not update data while you complete the transfer.

You'll need to communicate this downtime to application users and also the new URL that they need to access their application once you complete post-migration tasks on Oracle Visual Builder Studio. Typically, the URL part that changes is the part that identifies the host name where the Oracle Visual Builder and Oracle Visual Builder Studio instances are hosted, for example:

`https://`***`oci-oldinstance`***`/ic/builder/rt/visualapp/live/webApps/appname/`

and

`https://`***`oci-newinstance`***`/ic/builder/rt/visualapp/live/webApps/appname/`

For mobile apps that consume REST endpoints from custom business objects that you migrate to a new Oracle Visual Builder instance as the runtime, you'll need to update the appropriate app store with a new version of the mobile app that you have built using VB Studio. This requirement extends to any client application that consumes REST endpoints from business objects that you migrate to a new instance. You'll need to update these client applications so that REST calls continue to work post-migration.

Plan Your Migration

Review the following considerations when planning your migration from the Oracle Visual Builder instance to VB Studio.

- Verify the data center you use hosts VB Studio. This is important because VB Studio may not be available in the same data center regions as your Oracle Visual Builder instance. This means you need to identify other regions in which to run your instances. See https://cloud.oracle.com/data-regions.

- Confirm that you have supported Oracle Visual Builder instances that you can use as the runtime environments for your visual applications. You can't develop visual applications in VB Studio without one.
- Confirm that the new VB Studio instance and Oracle Visual Builder instances that you'll use as the development, test, and production environments are provisioned.
- Confirm that you can access your VB Studio instance.

Review the Migration Task Flow

Review the following list to understand the tasks that you or an administrator need to complete to migrate web and mobile applications from your Oracle Visual Builder instance to VB Studio by importing an archive.

The task list assumes you have provisioned the new VB Studio instance and Oracle Visual Builder instances that you'll use as the development, test, and production environments.

1. Review the entries for users and roles in Oracle Identity Cloud Service to ensure that the instance of Oracle Identity Cloud Service that the new instance uses matches the entries in the instance of Oracle Identity Cloud Service used by VB Studio. This task is not necessary if both old and new instances use the same instance of Oracle Identity Cloud Service. If you use a new instance of Oracle Identity Cloud Service, make sure that your application users are granted access to the new instance with the appropriate roles.
2. For each visual application that will be migrated, the project owner should create a new project in VB Studio and add other team members to the project, if any.
3. For each visual application (and for each version of each visual application that you want to migrate), migrate the visual application sources and data from the Oracle Visual Builder instance, either by exporting it as an archive and then importing it, or by cloning the visual application's Git repository.

   > **Note**
   >
   > It is not possible today to import a version of a visual application; import always creates a new visual application in the VB Studio project.

4. Ensure proper mapping of virtual roles to Oracle Identity Cloud Service groups and app roles. You might need to remap the virtual roles in your app's Settings editor if you are migrating an older app.
5. Re-enter the security-related details for your visual application that are not captured when migrated from the old instance. This includes any client IDs and authentication details, along with build configurations details needed to build mobile apps.
6. Update the visual application settings to include a root URL and version number, then merge these changes to your main branch in Git. Unlike apps created in VB Studio, those created in Visual Builder aren't configured with a root URL. You'll need to provide one for your visual application before you deploy it. You'll also want to specify a version number at the same time.
7. Configure the project's build jobs to use the new repository and to deploy to your development, test and production environments.
8. Configure the settings for each of the Oracle Visual Builder environments to ensure that when your application is deployed to the environment it can access required services.

9. Use the build tools in VB Studio to share and publish the visual application to your Oracle Visual Builder environments.
10. Test the behavior of the migrated visual application. To perform testing, you may want to export data from the visual application on the old instance and then import the data into your environments. When you complete testing, perform the following post-migration steps to finish the migration of your visual applications.
    If your application is embedded with an iFrame in an Oracle Cloud Application, update and test your Cloud Application to ensure your embedded application continues to function post-migration.
11. For each migrated visual application, lock the live visual application on the old instance.
12. For each migrated visual application, use the Data Manager screen to export the live data from the old instance.
13. For each migrated visual application, use the Deployments screen in VB Studio to import the live data to the Visual Builder instance you are using for production.
14. Inform end-user clients to use the URL for the new service instance and to update any bookmarks they may have.

    Note

    This applies to accessing apps through a browser and client applications that may access business object REST APIs hosted in the new instance.

15. Mobile apps that do not use business object REST APIs and bypass Oracle Visual Builder authentication proxy should continue working the same post-migration. Otherwise, build a new version of your mobile app on the new instance and submit it to the appropriate app store for distribution to end users.
16. Migration is complete. Delete the old Oracle Visual Builder instance if you are no longer using it.

3

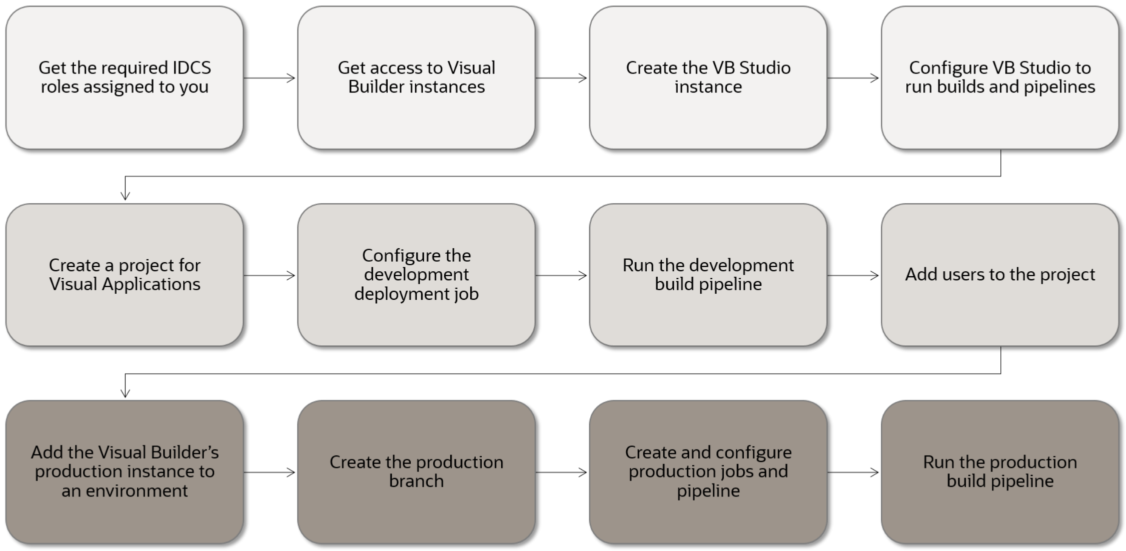
Migrate Your Visual Builder Visual Applications

Before you can migrate your visual application, an administrator needs to configure the Oracle Visual Builder Studio instance and an Oracle Visual Builder instance to make sure that your visual application can access the correct services and resources.

An administrator or project owner will also need to create the project that will contain your migrated visual application.

Set Up VB Studio for Developing Visual Applications

If your Oracle Visual Builder Studio (VB Studio) and Visual Builder instances are not already set up so you can create visual applications, you'll need to ask your administrator to set them up for you before you can migrate your visual application. An administrator can perform the following steps to prepare a project for your visual application.


Get the required IDCS roles assigned to you
Get access to Visual Builder instances
Create the VB Studio instance
Configure VB Studio to run builds and pipelines
Create a project for Visual Applications
Configure the development deployment job
Run the development build pipeline
Add users to the project
Add the Visual Builder's production instance to an environment
Create the production branch
Create and configure production jobs and pipeline
Run the production build pipeline

For a detailed description of these steps, see Set Up VB Studio for Developing Visual Applications in *Administering Visual Builder Studio*.

Migrate Your Visual Application Sources

After the project is set up for you, you can migrate your visual application sources from an Oracle Visual Builder instance by importing the application as an archive or by importing it from a Git repository.

The credentials for services used by the application are not imported. You'll need to supply the credentials after creating the new application from the imported sources. Similarly, the artifacts that are part of mobile build configurations (such as iOS provisioning profiles, keystores, and passwords) are not imported. You'll need to re-enter these details in the new application after you import it.

Export a Visual Application Archive from Oracle Visual Builder

In your old Oracle Visual Builder instance, use the Export action to create a ZIP archive of the visual application you want to migrate to a project in VB Studio.

When you export the application you can choose if you want the archive to include the development data contained in the application's custom business objects. If you choose not to include the data in your archive, you can export and import the data later.

Some information, such as credentials for external REST end points, is removed when you export an application. This information needs to be provided after the archive is imported.

To export a visual application and its custom business objects:

1. Open the Oracle Visual Builder instance.
2. On the Visual Applications home page, open the Application Options menu for the application version you want to export and select **Export**.

   If there are multiple versions of an application you must use the Options menu of the version that you want to export.

   Alternatively, when a visual application is open, you can choose **Export** in the application's options menu in the toolbar.
3. Click **Export with Data** in the Export Application dialog box.

   When you choose to export the application with data, the archive will include a json file (`entity.json`) and spreadsheet (`entity-data.csv`) for each custom business object. The json file describe the business object and the spreadsheet contains the business object data in the development database. If you choose to export the application without data, the archive will only contain the json file describing the business objects.

   The archive will always include the data for any business objects that are identified as containing Application Setup Data.

The visual application and its resources are exported as an archive file. The archive is saved to your local system in the location specified for your browser's downloads.

Import a Visual Application Archive into VB Studio

In your new VB Studio instance, you can create a new visual application by importing the archive of the visual application you exported from the Oracle Visual Builder instance.

To import a visual application archive:

1. On the Organization page in VB Studio, select the project where you want to import the visual application.

   Typically the project owner will have created the project for you and added you as a team member. Best practice is for a project to only contain one visual application.
2. Open the Designer page and click **+ Create Workspace**.
3. In the Create Workspace dialog, type a name for your workspace and click **Import from File**.
4. Drag your visual application archive file from your local system into the upload area. Alternatively, click the upload area in the dialog box and use the file browser to locate the archive on your local system.

5. In the Git Repository field, enter a repository name and working branch name for the Git repository that VB Studio will create when it creates the workspace.
6. In the Development Environment dropdown list, select the environment to use.
7. Click **Create Workspace**.

VB Studio creates a workspace with the name and the repository option you specified. Once VB Studio completes set up of the workspace, it opens the workspace where you can work with the visual application that you imported. After you import a visual application, you might need to provide additional details such as service credentials for the application you imported.

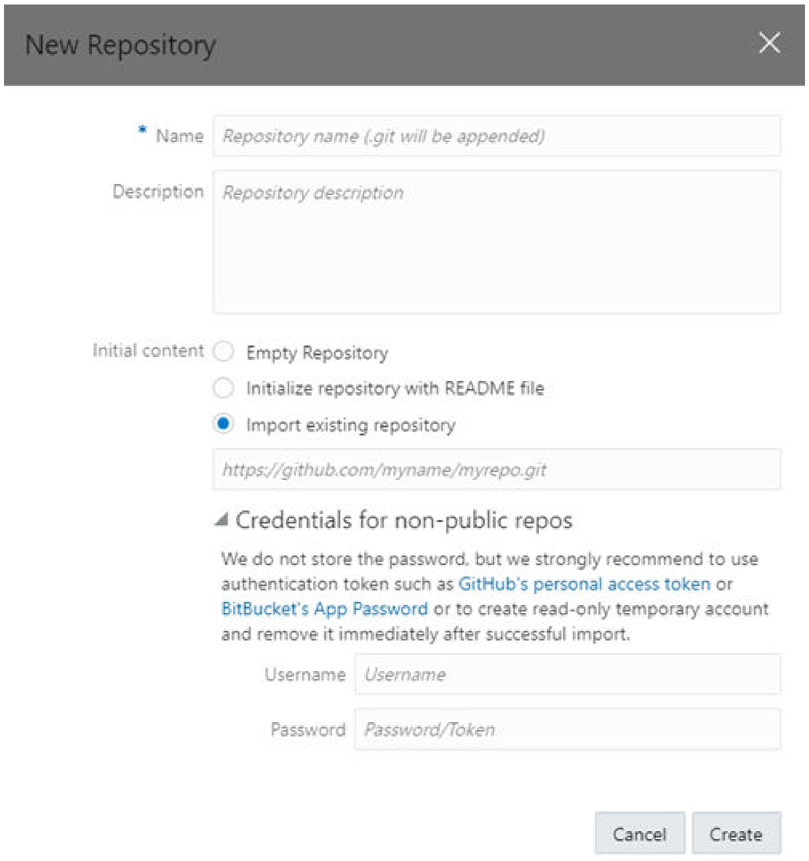
When you import the archive, VB Studio automatically performs some migration tasks for you, including creating a repository in the project containing the sources and data in the archive, and setting up a build pipeline that is configured to use the new repository and deploy the application to the environment you selected.

Import a Visual Application from a Git Repository

If your visual application's sources are stored in a Git repository, you can import the visual application sources by creating a copy of the repository in your project. You'll then need to create build jobs that use the new repository.

To import a visual application from an existing Git repository:

1. On the Organization page in VB Studio, select the project where you want to import the visual application.
2. On the Projects page, click **+ Create Repository** in the Repositories tab to open the New Repository dialog box.


New Repository
* Name
Repository name (.git will be appended)
Description
Repository description
Initial content
Empty Repository
Initialize repository with README file
Import existing repository
https://github.com/myname/myrepo.git
Credentials for non-public repos
Username
Username
Password
Password/Token
Cancel
Create

We do not store the password, but we strongly recommend to use authentication token such as GitHub's personal access token or BitBucket's App Password or to create read-only temporary account and remove it immediately after successful import.

3. In the dialog box, enter a name for the new repository and select **Import existing repository**.
4. Enter the https location of the repository containing your visual application and supply credentials, if needed. Click **Create**.
5. Open the Designer page and click **+ Create Workspace**.
6. In the Create Workspace dialog, type a name for your workspace and click **Clone from Git**.
7. Select the Git repository you just created and select the branch of the repository you want to clone. This can be master, or any other branch..
8. Select New branch from selected and type a name for the branch you want to create.
9. Select the Development Environment. (You might have only one option.) Click **Create Workspace**.

VB Studio creates a workspace with the name and the repository option you specified. Once VB Studio completes set up of the workspace, it opens the workspace where you can work with the visual application that you imported. After you import a visual application, you might need to provide additional details such as service credentials for the application you imported.

In this case, VB Studio doesn't create any build jobs for you when you create your workspace because the Git repository already exists in your project.

Update Your Visual Application Settings

After you import your visual application into a Oracle Visual Builder Studio project, update the application settings with a root URL and app version.

Unlike apps created in VB Studio, those created in Oracle Visual Builder aren't configured with a root URL. You'll need to provide one for your app before you deploy it. You'll also want to specify a version number at the same time.

You can provide the root URL and version number through the Application tab of the Settings editor or update both by opening the `visual-application.json` file from the Source View tab in the Navigator and editing it directly.

The root URL and version are used to generate a unique URL for the app when you deploy it. For example, a root URL of "MyVBApp" and a version of "1.0" results in a URL of:

```
https://<host>/MyVBApp/1.0/index.html
```

To provide a root URL and version number for your imported app:

1. From your application's workspace, click the Menu option in the upper right corner of the application's workspace toolbar, then select **Settings**.
2. In the Settings editor's General tab, enter a version number in the **Version** field.

   You can use any numbering scheme that makes sense to you.
3. Enter a unique name for your app in the **Root URL** field.

   This name is used as the root URL for the app when it is deployed. Don't use a root URL that is already used by other apps on the Visual Builder instances that you'll deploy to.

These setting are saved in the `visual-application.json` file for your app. You can open and edit this file from Source View.

Before you can deploy your visual application, you'll need to push these changes to your Git repository and merge them to the main branch. See Manage Your Visual Applications With Git in *Building Responsive Applications with Visual Builder Studio*.

Managing the Build Pipelines for Your Environments

You use build pipelines in VB Studio to package and deploy applications to the Oracle Visual Builder instances that provide the runtime environments for development, testing and production.

After importing sources into your project, you or the project owner will need to create and configure the pipelines to use the sources in your new Git repository, and also to configure the build options and settings for each pipeline.

- When you import an application archive in the Create Workspace dialog box, VB Studio creates a pipeline to package and deploy the app to the development environment when it creates the new Git repository for the app in the project.
- If you create a Git repository by cloning another repository, you'll need to create a pipeline to package and deploy the app to the development environment. See Configure the Packaging Job and Configure the Deployment Job in *Administering Visual Builder Studio*.

You'll also need to create and configure pipelines for any other Oracle Visual Builder instances you'll use, for example, the production environment for your app. After an administrator has set up the environments, you or the project owner can create pipelines for them. See Set Up the Project to Deploy for Production in *Administering Visual Builder Studio*.

Manage the Settings of Your Visual Builder Environments

When you publish a visual application, the build pipelines deploy the app to an Oracle Visual Builder instance that provides the app's runtime environment. There are some settings that you or an administrator need to set in each of your instances so that your applications run correctly.

You can manage some of your runtime instance settings from the app's Settings editor in VB Studio, but some settings need to be configured in the Tenant settings of each of your Oracle Visual Builder instances.

To manage these settings you'll need to log in to the Oracle Visual Builder instance as an administrator to access the Tenant settings. You'll need to configure the Tenant settings to match the settings of your old Oracle Visual Builder instance. For details on setting up environments to work with your visual application project, see Set Up the Project to Deploy for Production in *Administering Visual Builder Studio*.

Access Tenant Settings

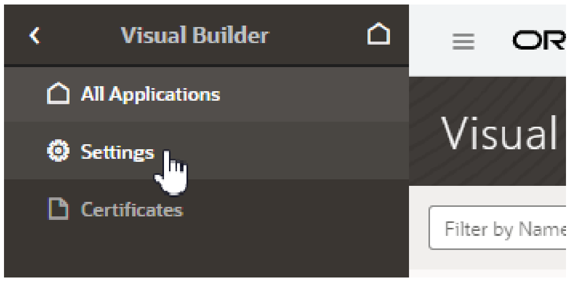
An instance administrator can access the Tenant Settings page for managing the instance’s global settings from any Visual Builder page.

The Tenant Settings page contains three tabs: General, Tenant Database and Services. The General tab has panels for configuring security settings, specifying Access Denied messages, and configuring the Component Exchange details. You use the Tenant Database tab to switch to an Oracle database and to see how much database space your applications are using. You use the Services tab to add and edit the backend services that are accessible to apps in the tenant.

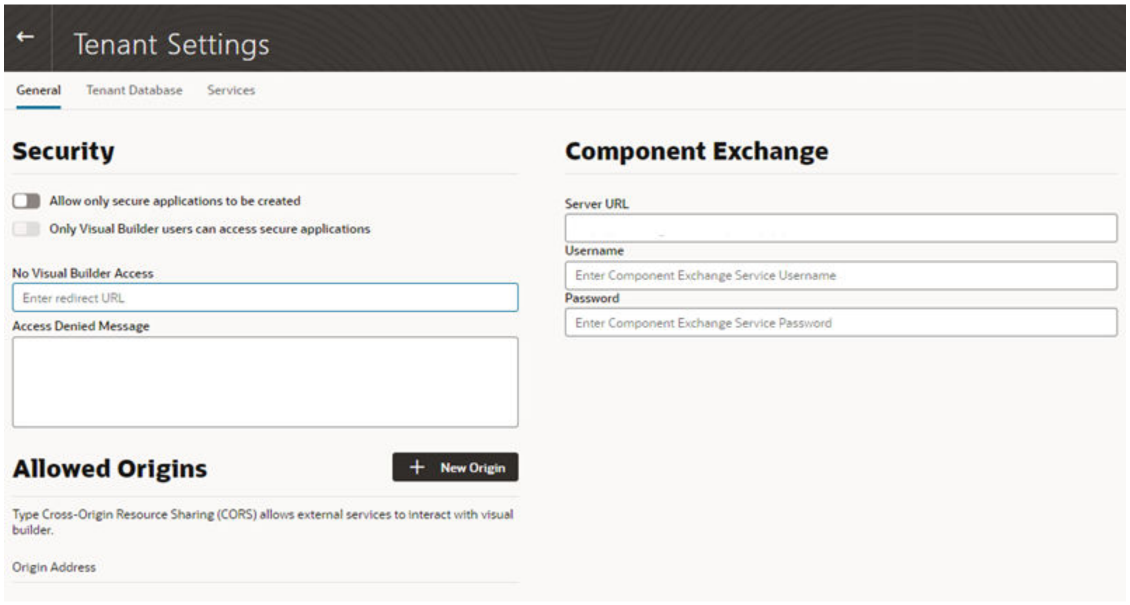
To open an instance’s Tenant Settings page:

1. In the upper-left corner of the Visual Builder title bar, click **Navigation Menu** ☰.
2. Click **Settings** in the main menu.

   If you are developing visual applications, open the main navigation pane on the Home page and select **Settings**.



The settings available for the instance are grouped in the General, Tenant Database and Services tabs on the page.


Tenant Settings

General | Tenant Database | Services

Security

Allow only secure applications to be created

Only Visual Builder users can access secure applications

No Visual Builder Access: Enter redirect URL

Access Denied Message

Allowed Origins — New Origin

Type Cross-Origin Resource Sharing (CORS) allows external services to interact with visual builder.

Origin Address

Component Exchange

Server URL

Username: Enter Component Exchange Service Username

Password: Enter Component Exchange Service Password

Configure Security Options for Applications

Administrators can use the Security panel in the Tenant Settings page to require authentication for all applications in the instance.

When an administrator enables the **Allow only secure applications to be created** option, all published and staged applications in the instance will require user authentication. When the option is enabled, users must log in to access the applications in the instance, even if anonymous access is allowed in the application's settings. When the option is not enabled, applications can be created that allow access to anonymous users.

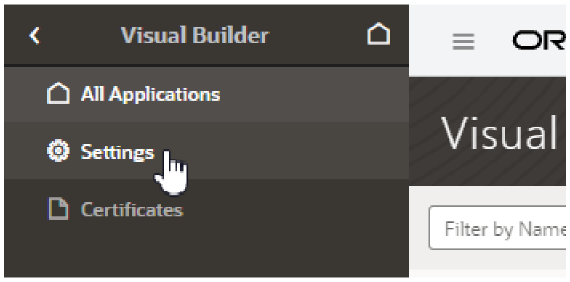
When an application has the default security settings, any user with a valid login can access the pages in an application. A developer can modify the default security settings to define the roles that can access applications, pages and components.

When the secure application option is enabled, an administrator can enable the **Only Visual Builder Users can access secure applications** option so that only Visual Builder users (those assigned the default Service User role) can access the staged and published applications in the instance. For example, this allows you to configure security so that users assigned the Visual Builder Developer role can access the designer, but can’t access the published application and data because they are not assigned the Visual Builder Service User role.

An administrator can also use IDCS roles when configuring the instance's security so that a user's access is limited to just the secure applications. Users assigned the selected IDCS role would be able to access the applications, but would be prevented from accessing Visual Builder or Oracle Integration resources external to the application, such as other Oracle Integration integrations.

To configure the security options for all applications in the instance:

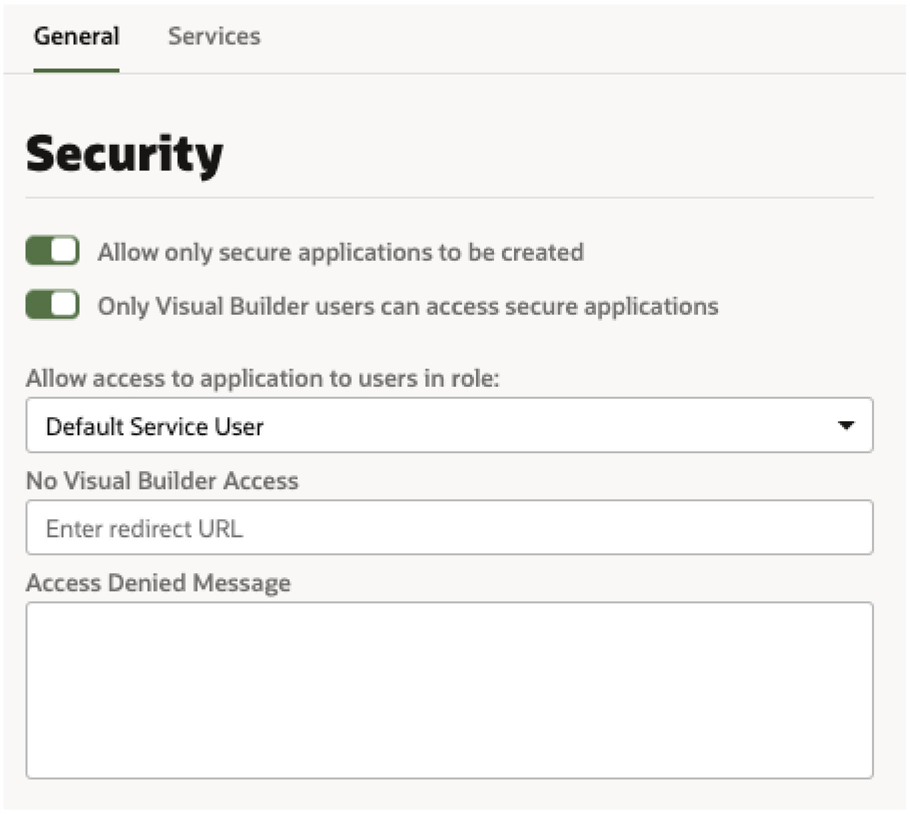
1. In the upper-left corner of the Visual Builder title bar, click **Navigation Menu** ☰.
2. Click **Settings** in the navigation menu to open Tenant Settings.



3. In the **Security** panel, enable **Allow only secure applications to be created**.

   Anonymous users can’t access the applications when this secure applications option is enabled.
4. Select the **Only Visual Builder Users can access secure applications** option if you want to allow only Visual Builder users (users assigned the Service User role) access to the applications.

   To change the users allowed to access the application to those assigned a specific IDCS role *instead of* those assigned the default Service User role, select the IDCS role in the dropdown list under **Allow access to application to users in role**. This option is only available when both of the other security options are enabled.



Security

Allow only secure applications to be created

Only Visual Builder users can access secure applications

Allow access to application to users in role:

Default Service User

No Visual Builder Access

Enter redirect URL

Access Denied Message

5. Specify what users denied access to the secure application will see:
   a. Enter the URL you want the users redirected to when they can't access the app.
   b. Enter an Access Denied message that they will see when denied access to a page in the app.

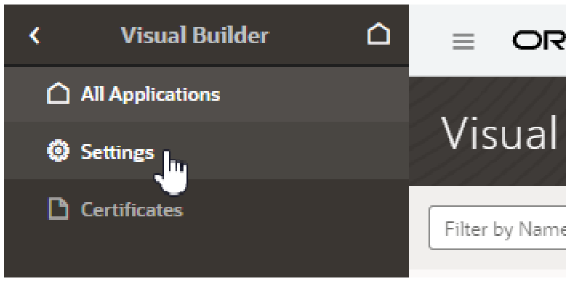
Set Page Messages for Access Denied Errors

Administrators can use the instance’s settings page to specify a URL that users are navigated to when they are denied access to an application or page.

Authenticated users might see an Access Denied page or message when they attempt to access an application or page in an application that their user role is not permitted to access. Administrators can set the default page or message that users see when they are denied access to an application or page. Access Denied messages that are set at the application level in the General Settings of an application will override messages set in the instance’s settings page. The default Access Denied page and message is used if the message options in this panel are not set.

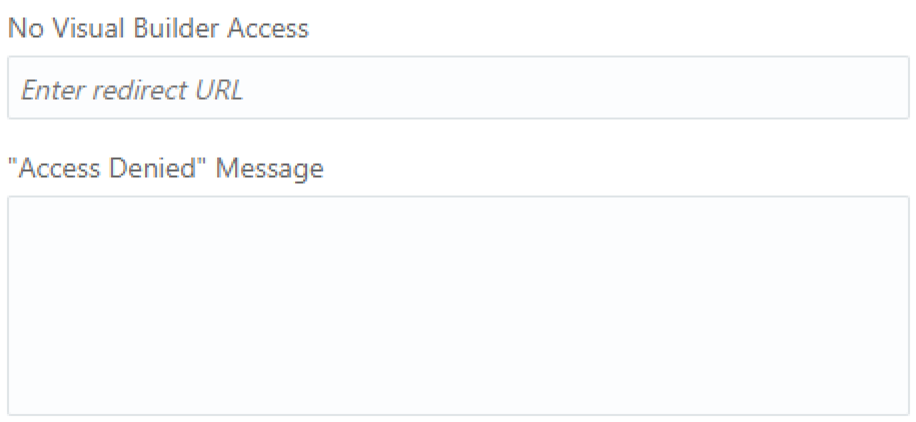
To specify an Access Denied page or message for applications in the instance:

1. In the upper-left corner of the Visual Builder title bar, click **Navigation Menu** ☰.
2. Click **Settings** in the navigation menu to open Tenant Settings.



3. In the **Security** panel, type a URL that users are directed to when denied access to an application.

   The URL that you specify is used as the Access Denied page for all applications in the instance and should be accessible to users who are not logged in.



> **Note**
>
> If you are configuring settings for classic applications, the Access Denied settings are set in the **Messages** panel.

4. Type the message that you want users to see when they are denied access to a page.

   The message that you enter will be displayed in the Access Denied page for all applications in the instance except for those where a message was set at the application level in the application's General Settings page.

Allow Other Domains Access to Services

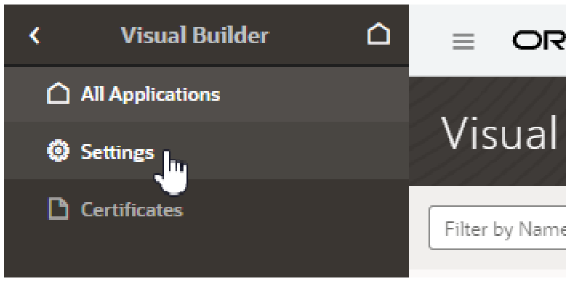
Use the Global Settings page to specify the domains that are permitted to interact with services in your instance.

Cross-Origin Resource Sharing (CORS) is a mechanism that enables you to specify the domains that are allowed to exchange data with applications in your instance. By default, incoming requests from domains not on your instance's list of allowed origins are blocked from accessing application resources.

To add a domain to the list of allowed origins:

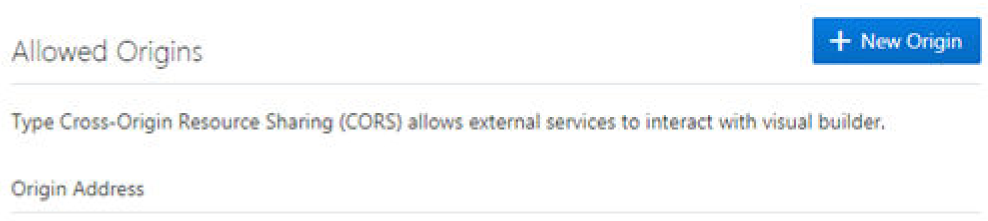
1. In the upper-left corner of the Visual Builder title bar, click **Navigation Menu** ☰.

2. Click **Settings** in the navigation menu to open Tenant Settings.



3. In the **Allowed Origins** panel, click **New Origin** and type the URL of the domain that you want to allow. Click **Submit**.

   The URL must be a fully-qualified domain, meaning it must contain `http://` or `https://`, for example, `https://myoracle.cloud.service`. You must explicitly enter each fully-qualified domain that you want to allow. To allow both `http://` and `https://` connections from a domain, you would need to add both domains (`https://myoracle.cloud.service` and `http://myoracle.cloud.service`).



Type Cross-Origin Resource Sharing (CORS) allows external services to interact with visual builder.

The Allowed Origins panel lists all origins that are permitted to retrieve information from the instance.

Switch to Your Own Oracle DB Instance

The database provisioned with your Visual Builder instance is used to store data for your business objects and your app's metadata, but this database has a 5GB limit and you can't access the data in the objects using regular SQL.

If the 5GB limit is insufficient for your tenant schema, you can configure your instance to use an Oracle DB instance that has more space instead of the default database. You can connect to an Oracle DBaaS or Autonomous Transaction Processing (ATP) database instance. Using an ATP database will give you more space and direct SQL access to the objects VB creates. You can also use a Free Forever Oracle ATP, which provides 20GB of storage for free.

To use a different Oracle DB instance, you use a wizard in the Tenant Settings to create a connection to the database instance and export the applications stored in the tenant's current database.

If you decide to use JDBC to connect to your DBaaS instance, you must include the privileges required to enable the ADMIN user to create a tenant schema. The following SQL shows the grants that are needed:

```
CREATE USER [adminuser] IDENTIFIED BY [password];
GRANT CONNECT, RESOURCE, DBA TO [adminuser];
GRANT SELECT ON SYS.DBA_PROFILES TO [adminuser] WITH GRANT OPTION;
GRANT SELECT ON SYS.DBA_USERS TO [adminuser] WITH GRANT OPTION;
GRANT SELECT ON SYS.DBA_DATA_FILES TO [adminuser] WITH GRANT OPTION;
GRANT SELECT ON SYS.DBA_SEGMENTS TO [adminuser] WITH GRANT OPTION;
```

If you decide to use ATP, you'll need to include the `wallet.zip` file in the wizard in addition to the connection info. You might want to create a new ATP ADMIN user with the correct admin privileges. The following SQL statement shows how to create a second ATP ADMIN user in SQL*Plus or SQL Developer.

```
DROP USER [adminuser] CASCADE;
CREATE USER [adminuser] IDENTIFIED BY [password];
GRANT CREATE USER, ALTER USER, DROP USER, CREATE PROFILE TO [adminuser] WITH
ADMIN OPTION;
GRANT CONNECT TO [adminuser] WITH ADMIN OPTION;
GRANT RESOURCE TO [adminuser] WITH ADMIN OPTION;
GRANT CREATE SEQUENCE, CREATE OPERATOR, CREATE SESSION,ALTER SESSION, CREATE
PROCEDURE, CREATE VIEW, CREATE JOB,CREATE DIMENSION,CREATE INDEXTYPE,CREATE
TYPE,CREATE TRIGGER,CREATE TABLE,CREATE PROFILE TO [adminuser] WITH ADMIN
OPTION;
GRANT UNLIMITED TABLESPACE TO [adminuser] WITH ADMIN OPTION;
GRANT SELECT ON SYS.DBA_PROFILES TO [adminuser] WITH GRANT OPTION;
GRANT SELECT ON SYS.DBA_USERS TO [adminuser] WITH GRANT OPTION;
GRANT SELECT ON SYS.DBA_DATA_FILES TO [adminuser] WITH GRANT OPTION;
GRANT SELECT ON SYS.DBA_SEGMENTS TO [adminuser] WITH GRANT OPTION;
```

Note

If you get an error `Failed to verify the target database` in the Change Tenant Database dialog when switching the database, it might be because you don't have the required privileges, or because the database is not reachable. (Visual Builder cannot reach databases in private subnets, except when Visual Builder is provisioned as a private endpoint in the same private subnet as the database.)

If you see the error, confirm that the ADMIN user (`adminuser`) has the required privileges. You might also need to assign the SYSOPER and SYSDBA roles to the ADMIN user:

```
GRANT SYSOPER, SYSDBA TO [adminuser];
```

You can run the following query to confirm the ADMIN user has the necessary privileges:

```
select * from v$pwfile_users;
```

In the wizard you need to select and export all the applications in your instance that you want to keep. After confirming that your instance is using the new database instance, you must import the exported applications into Visual Builder to save them in the new database instance.

Note

If you have live applications already on the instance:

- Before switching to a new database, make sure to backup the data in their business objects using the export options in the Visual Builder data manager. You'll then be able to import that data back into the new apps you'll create from the application archives you export in the wizard.
- Lock the live applications before changing the settings of your instance's database to prevent users from using them during the migration process. You can unlock the applications when the migration process is finished. You lock and unlock live applications in the Application Options menu on the Visual Builder Home page.

To switch to a different Oracle DB instance:

1. Open the Tenant Database tab.

   You can open your instance's Tenant Database tab from the instance Home Page, or by entering the URL directly in the browser window. It might be quicker to enter the URL directly if there is a problem loading the Home Page, for example, if the wallet is expired.

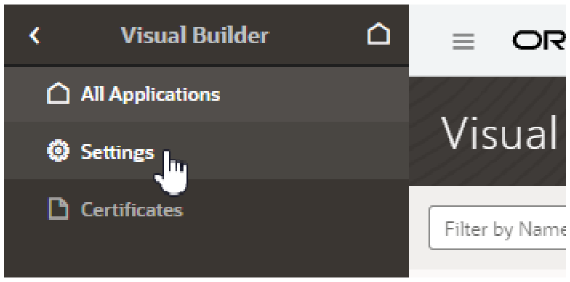
   - To open the Tenant Database tab using a URL, type the following in the browser's URL field:

     ```
     https://<instance-url>/ic/builder?root=settings&settingsSection=tenant-
     database
     ```

     In the URL above, replace <*instance-url*> with your instance's URL.
   - To open the Tenant Database tab from the Home Page:

     a. On the Visual Builder Home Page, click **Navigation Menu** ≡ in the upper-left corner of the Visual Builder title bar.

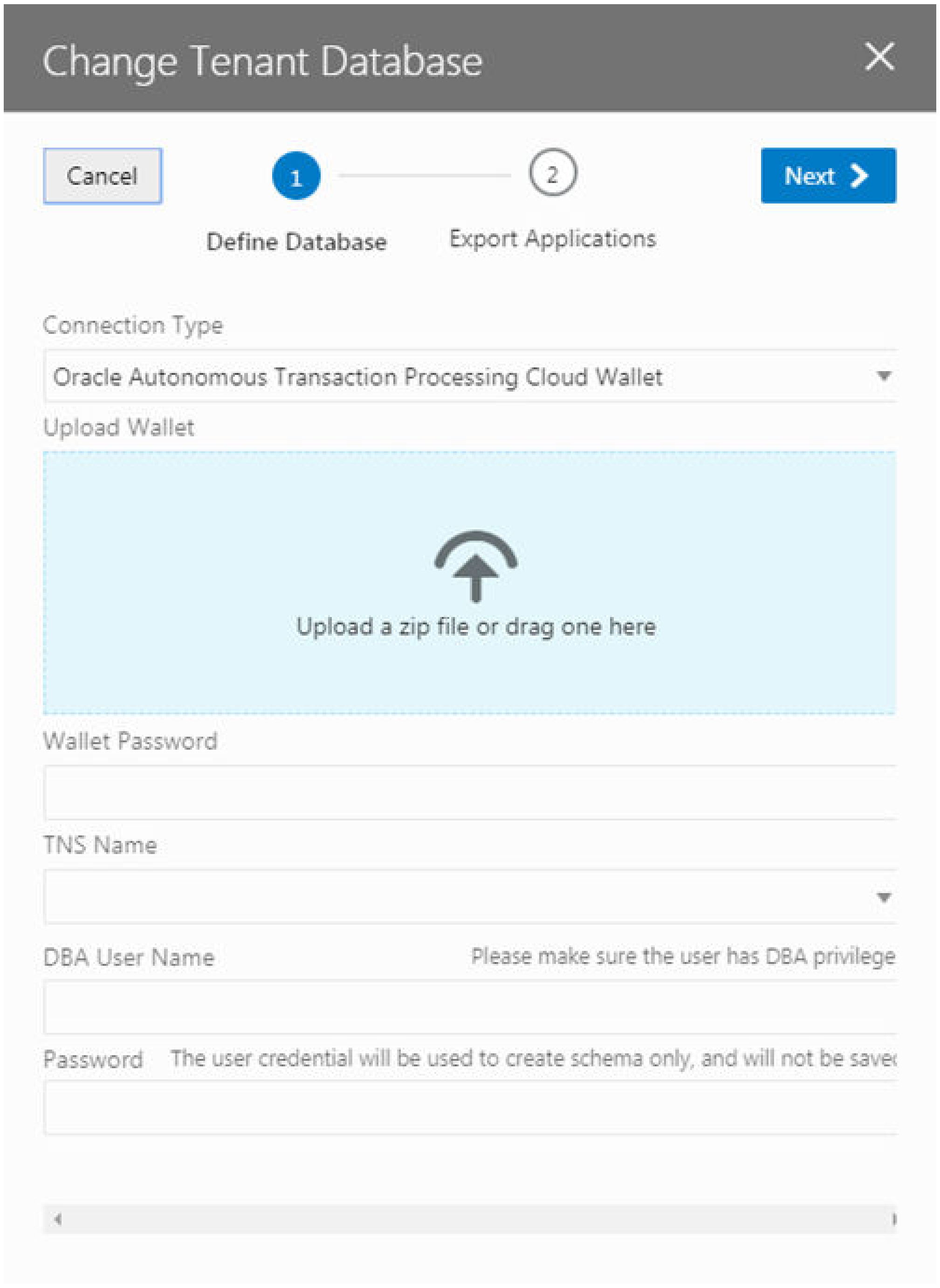
     b. Click **Settings** in the navigation menu to open Tenant Settings.

     c. Open the Tenant Database tab.

2. In the Tenant Database tab, click **Use Different Database** in the Tenant Database panel to open the Change Tenant Database wizard.

   In the Change Tenant Database wizard you supply the details for the connection to your Oracle DB instance.



Connection Type
Oracle Autonomous Transaction Processing Cloud Wallet
Upload Wallet
Wallet Password
TNS Name
DBA User Name
Please make sure the user has DBA privilege
Password
The user credential will be used to create schema only, and will not be save

3. Select a Connection Type in the drop-down list.

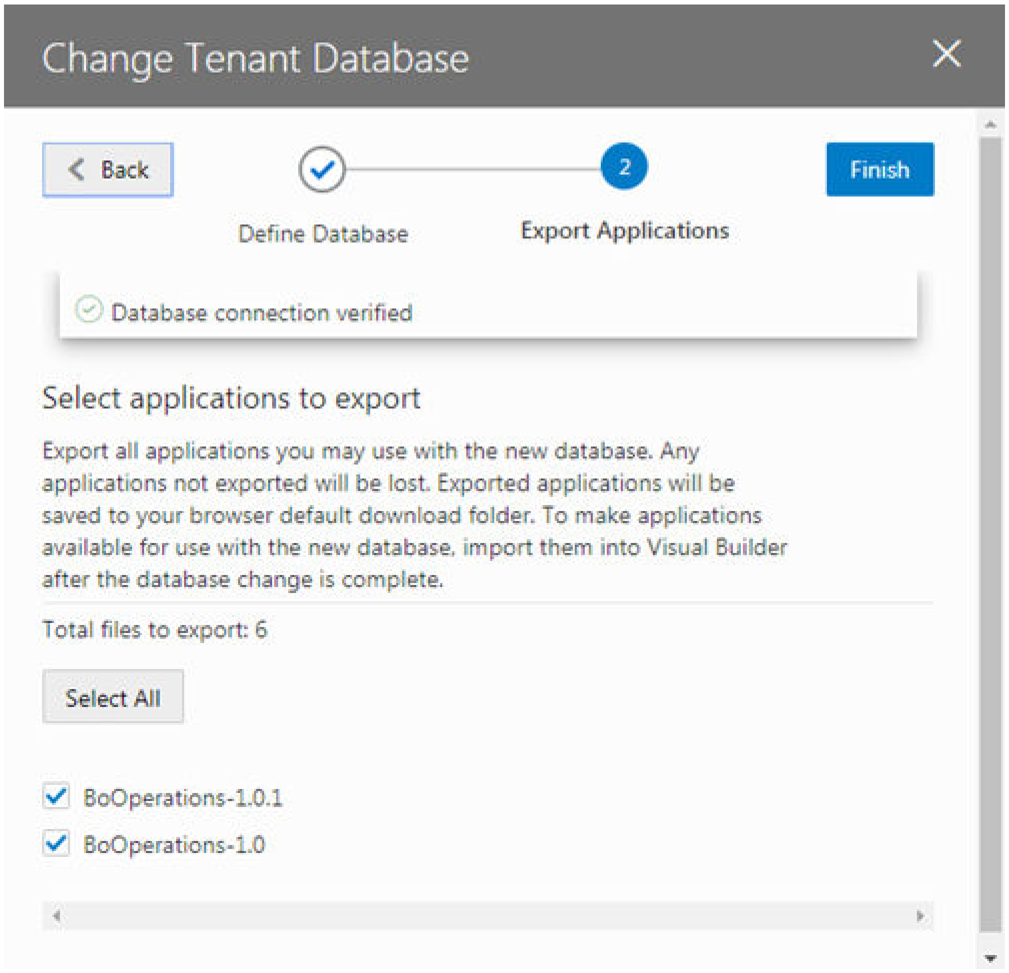
   You can connect to your Oracle DB instance using either JDBC or an ATP wallet.

4. Provide the details for connecting to your database. Click **Next**.

   The details you need to provide will depend upon the type of connection you selected.

5. Select all the applications that you want to export. Click **Finish**.

   You must select and export all the applications that you want to keep. Any applications that are not exported will be lost.



Database connection verified

Select applications to export

Export all applications you may use with the new database. Any applications not exported will be lost. Exported applications will be saved to your browser default download folder. To make applications available for use with the new database, import them into Visual Builder after the database change is complete.

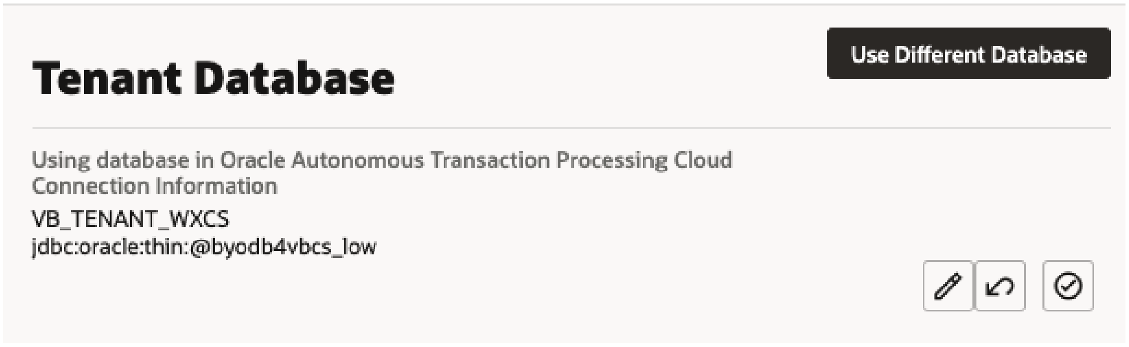
Total files to export: 6

Select All

- [x] BoOperations-1.0.1
- [x] BoOperations-1.0

When you click Finish, the applications that you selected are downloaded to your local file system. Exported application archives include the details about the application's user roles, and they will be available when you re-import your app into the new database.

After switching the database, the Tenant Database pane displays the connection information for your tenant's database. In the following image you can see that the instance is now using an Autonomous Transaction Processing (ATP) database instance.



Using database in Oracle Autonomous Transaction Processing Cloud
Connection Information
VB_TENANT_WXCS
jdbc:oracle:thin:@byodb4vbcs_low

Note

If you decide to revert back to using the embedded database, you can click ↶ in the Tenant Database pane. You'll be prompted to confirm that you want to switch to using the instance's embedded database instead of the current one.

When you revert to using the embedded database, the visual applications in your current database are not transferred automatically. You need to export the apps you want to keep before switching the database, and then import them into the embedded database.

Visual Builder automatically manages the schemas and tables it uses for apps and business objects in your new DB, so you don't need to do anything further.

If you would like to access the business objects using SQL, you'll find that VB creates users/schemas with names that start with `VB_` followed by randomly generated strings. By examining the data dictionary you'll be able to find the users that represent specific apps. Note that you'll see separate schemas for dev, stage, and published instances of an app. The schemas for the dev and test instances will be re-created with different names with every new version of the app that you create. If you want to prevent the schema name for a published app from changing, when you publish new versions of the app you should choose the option to not replace the data.

Note

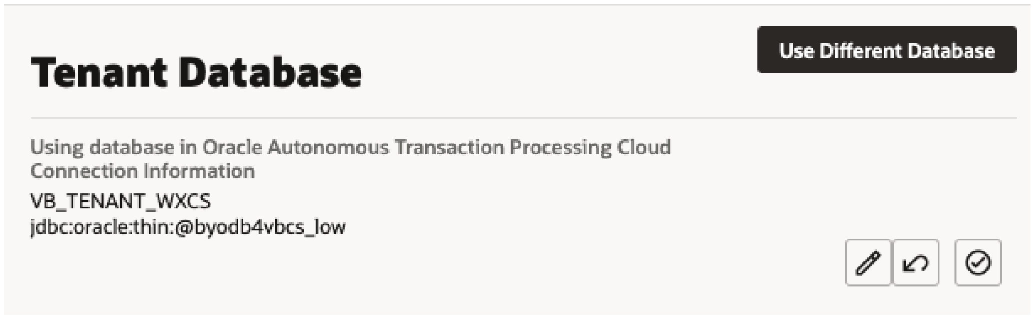
Instead of having Visual Builder create and manage schemas, you can make a schema that already exists in your database available to applications, so developers can create business objects based on existing DB tables and views. If you choose to use your own schema, make sure you understand the requirements and limitations when using your own schema. For details, see Switch to Your Own Database Schema for Business Objects in *Developing Applications with Oracle Visual Builder*.

If you use your own schema, only one schema is used for the app's dev, staged, and published instances. See Make Schemas in an Oracle DB Instance Available to Applications.

Switch From One ATP Database to Another

It's not possible to switch from one ATP database to another directly, so you'll first need to switch from your ATP database back to the embedded database. You can then use the Change Tenant Database wizard to switch from the embedded database to the new ATP database.

1. In Visual Builder, export each of your visual applications and save them to your local system.
2. Revert to the embedded database.
   a. Open the instance’s Tenant Settings page, and then open the Tenant Database tab.
   b. Click ↶ in the Tenant Database pane to revert to the embedded database.



Tenant Database

Use Different Database

Using database in Oracle Autonomous Transaction Processing Cloud
Connection Information
VB_TENANT_WXCS
jdbc:oracle:thin:@byodb4vbcs_low

c. When prompted, confirm that you want to switch to the embedded database.

3. Switch to the new ATP database.

   Follow the steps in Switch to Your Own Oracle DB Instance above to switch from the embedded database to your new ATP database.

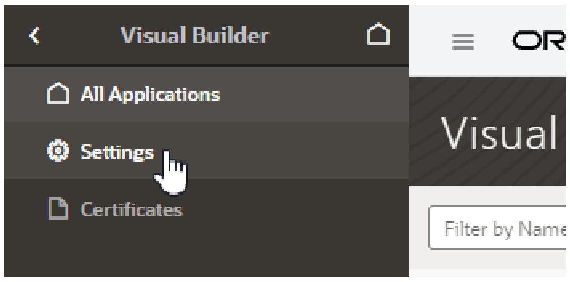
4. Import the applications you saved to your local system into the new ATP database.

Make Schemas in an Oracle DB Instance Available to Applications

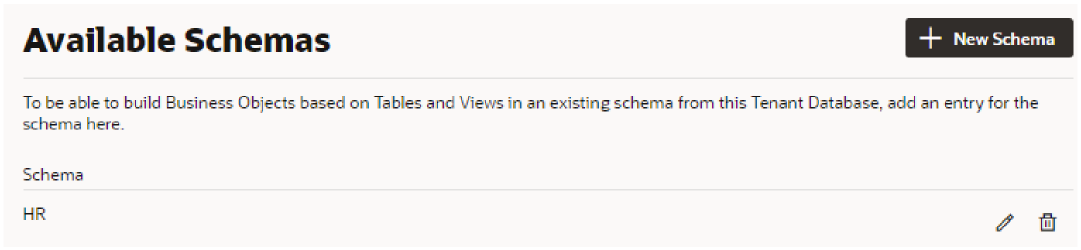
When you connect an Oracle database instance with your Visual Builder instance, application developers can use schemas predefined in the tenant database to create business objects based on existing tables and views for an application. But for developers to access these schemas, you'll first need to make them available to applications.

To make a tenant database's existing schema available to applications:

1. In the upper-left corner of the Visual Builder title bar, click **Navigation Menu** ☰.
2. Click **Settings** in the navigation menu to open Tenant Settings.



3. Click **+ New Schema** in the Available Schemas panel.



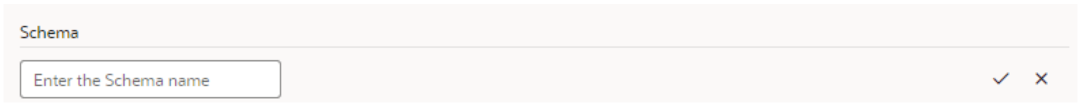
Available Schemas

+ New Schema

To be able to build Business Objects based on Tables and Views in an existing schema from this Tenant Database, add an entry for the schema here.

Schema
HR

4. Enter a name for the schema and click the check mark icon.


Schema

Enter the Schema name

After the schema is added, you can edit its name or delete it entirely, but remember any changes you make might break applications that use the schema.

Schema that exists in the tenant database and has been added to the list of available schemas will become available for selection in an application's Settings editor (under Schema Selection in the Business Objects tab).

Update Your ATP Wallet and Reset an Expired Password

If you switch to your own Oracle DB instance and the credentials you use to access the instance expire, you can use the Update Tenant Database Connection dialog box to update your ATP wallet and renew expired credentials.

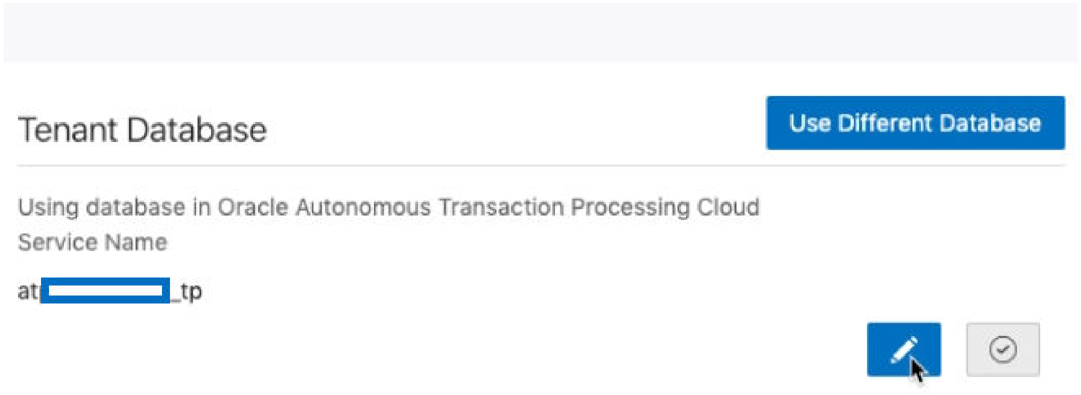
To regenerate the expired values, you need to provide the ADMIN user credentials that you provided when you first switched to your own Oracle DB instance. Visual Builder uses the ADMIN user credentials to generate new Visual Builder tenant credentials to replace the expired credentials. Visual Builder does not store the ADMIN user credentials that you supply.

1. Open the General tab of the instance's Tenant Settings page.

   If you cannot navigate to the Tenant Settings page from the navigation menu, you can open the page directly by entering the page's URL in the browser. The URL will be similar to

   `https://<Instance-URL>/ic/builder/?root=settings&settingsSection=tenant-database`.

2. In the Tenant Database field, click the Edit icon to open the Update Tenant Database Connection wizard.

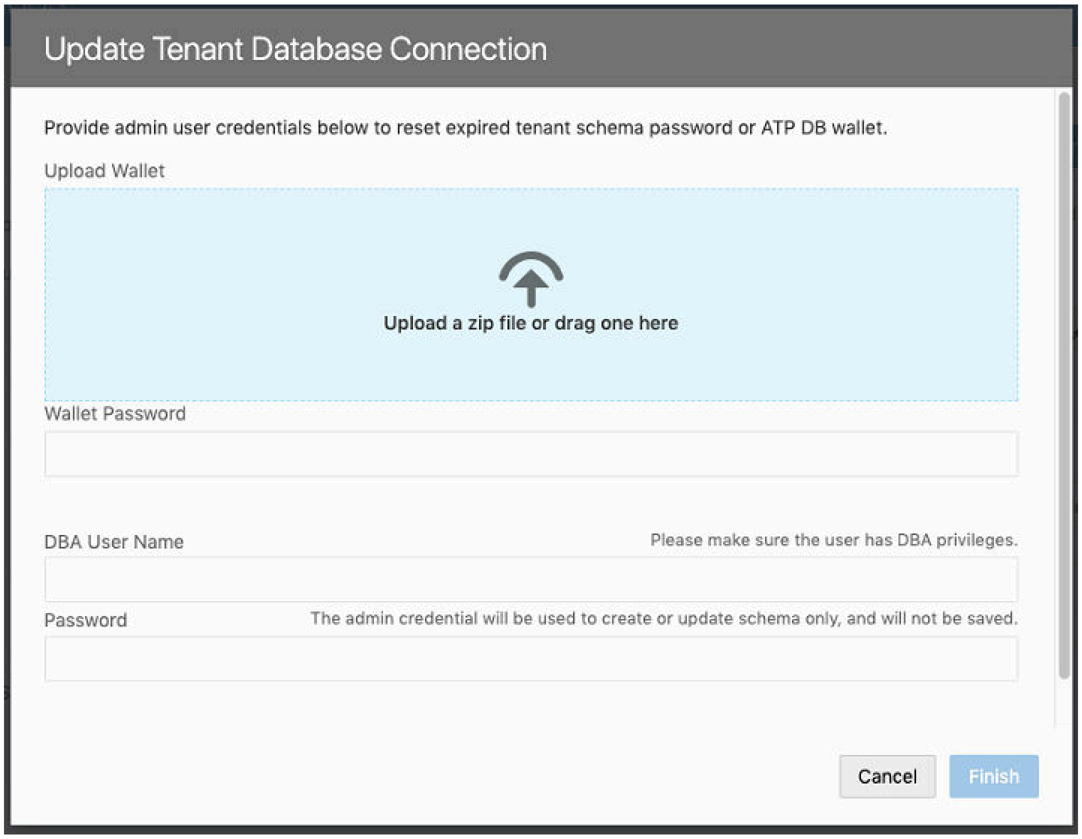
   Tenant Database

   Using database in Oracle Autonomous Transaction Processing Cloud

   Service Name

   at[redacted]_tp

3. In the Update Tenant Database Connection wizard:
   - Reset expired credentials by supplying the ADMIN user credentials and ATP wallet that Visual Builder will use, or
   - Update the wallet for your Oracle DB instance by uploading the new ATP wallet.


Update Tenant Database Connection

Provide admin user credentials below to reset expired tenant schema password or ATP DB wallet.

Upload Wallet

Upload a zip file or drag one here

Wallet Password

DBA User Name — Please make sure the user has DBA privileges.

Password — The admin credential will be used to create or update schema only, and will not be saved.

4. Click **Finish**.

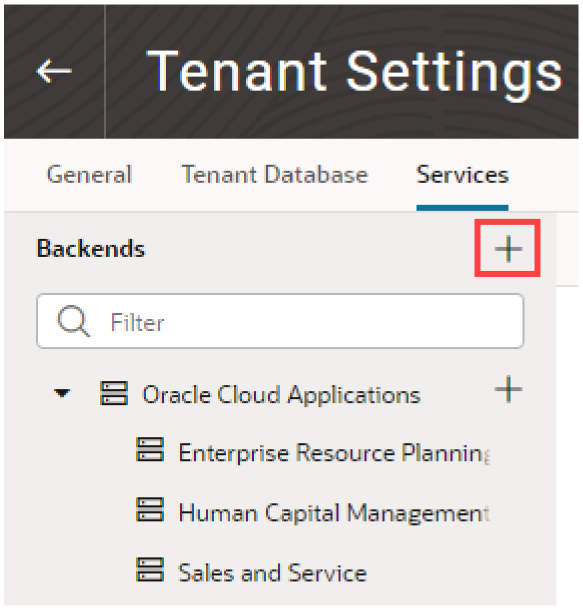
Add a Connection to Integration Applications

Administrators can use the Services tab in the Tenant Settings page to add a connection to an instance of Oracle Integration as a backend service.

If you are using multiple Visual Builder instances, for example, development and production instances, you might need to add connections to Oracle Integration in more than one instance.

To add a connection to an Oracle Integration instance:

1. Open the instance's Tenant Settings page.
2. In the Services tab, click **Create Backend** and choose **Integrations** in the Create Backend dialog.



3. In the dialog, type the Server URL of the backend service, configure other settings such as security as needed, and click **Create**.

Add a Connection to Oracle Cloud Applications

The list of REST services in the service catalog of a visual application is retrieved from an Oracle Cloud Applications backend service. Specify the instance URL of the Oracle Cloud Applications backend service in the Tenant Settings page.

All visual applications in the tenant will use the Oracle Cloud Applications instance URL specified in Tenant Settings, but a visual application can be configured to use a different Oracle Cloud Applications backend service by specifying a different instance URL in the Backends tab (which you access from the Navigator's Services tab). The tenant-level backend configuration is ignored if you or a visual application developer configures a different Oracle Cloud Applications backend service in a visual application’s Backends tab.

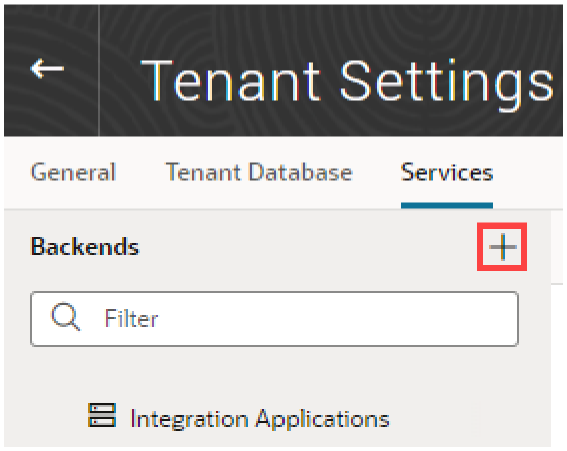
The authentication choices available to configure a tenant-level Oracle Cloud Applications backend are:

- Basic Auth: Uses a fixed username and password for authentication.
- Oracle Cloud Account: Needs federation between Oracle Cloud Applications and Visual Builder.
- Delegate Authentication (previously called Propagate Current User Identity): Same as Oracle Cloud Applications. That is, it needs federation between Oracle Cloud Applications and Visual Builder.
- None: This assumes your Oracle Cloud Applications REST API can be called without any authentication, which is not usually the case.

If the necessary prerequisites for setting a tenant-level Oracle Cloud Applications backend service are not available, then a visual application developer can set up a backend service at the visual application level where more options are available. Another option is for you (the service administrator) to configure the Oracle Cloud Applications backend with `None` and let the visual application developer override the authentication setting at the visual application level.

To specify an Oracle Cloud Applications service for the tenant:

1. Open the instance's Tenant Settings page.
2. In the Services tab, click **Create Backend**, then choose **Oracle Cloud Applications** in the Create Backend dialog.



   When specifying the URL in the Tenant Settings, you (the service administrator) only need to provide the instance URL of the Oracle Cloud Applications backend service to retrieve the list of services.
3. In the dialog, type the Server URL of the backend service, and configure other settings, such as security, as needed.
4. (Optional) After you configure settings for the backend, add headers to the backend.

   Backend headers that you add will be applicable for any service connection to this backend, irrespective of the server or application profile that is used.
5. Click **Create**.

   Visual Builder automatically discovers the interfaceCatalogs endpoint of the Oracle Cloud Applications backend, which retrieves the list of services and their metadata. This endpoint is typically in the form:

```
https://<My Oracle Cloud Applications Instance URL >/helpPortalApi/
otherResources/latest/interfaceCatalogs
```

   This endpoint is publicly accessible without any authentication.

   If there is a problem creating the connection, verify the instance URL of the Oracle Cloud Applications instance.

Add a Connection to Process Cloud Service

Administrators can use the instance's Tenant Settings page to add a connection to an instance of Oracle Process Cloud Service as a backend service.

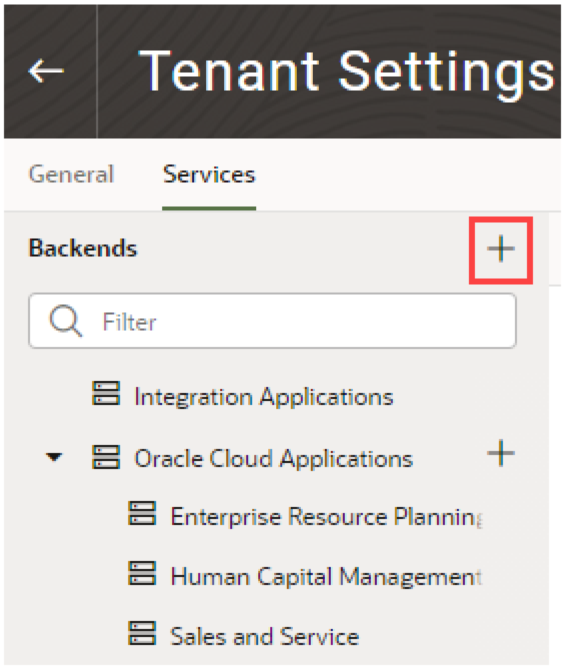
> **Note**
>
> Oracle Process Cloud Service is now deprecated; for details, see Deprecated Features.

To add a connection to an instance of Oracle Process Cloud Service as a backend service, the instance of Oracle Process Cloud Service should be co-hosted with Visual Builder because the authentication types that Visual Builder supports for this configuration is Oracle Cloud Account or Propagate Current User Identity. In most cases, this backend service (Oracle Process Cloud Service) will be preconfigured for your Visual Builder instance.

If you are using multiple Visual Builder instances, for example, development and production instances, you might need to add connections to Oracle Process Cloud Service in more than one instance.

To add a connection to an Oracle Process Cloud Service instance:

1. Open the instance's Tenant Settings page.
2. In the Services tab, click **Create Backend** and choose **Process** in the Create Backend dialog.



3. In the dialog, type the Server URL of the backend service, configure other settings, such as security, as needed, and click **Create**.

Manage Self-signed Certificates

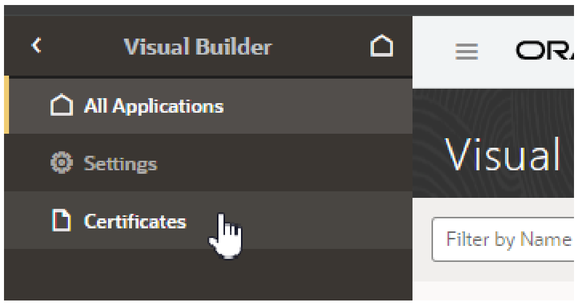
Administrators can use the Certificates page to upload and manage the self-signed certificates used by the instance to enable inbound and outbound SSL communications to a service's REST APIs

When creating connections to REST services that use self-signed certificates, you might need to add an API's certificate to your Visual Builder instance to validate SSL connections to that service. You can use the Certificates page to upload and remove certificate files (`.pem`) for services. Uploading a service's certificate file to the keystore will allow all applications in the instance to communicate with that service. The Certificates page displays a list of certificates that have been added. You can click the Delete button in a row to remove the certificate.

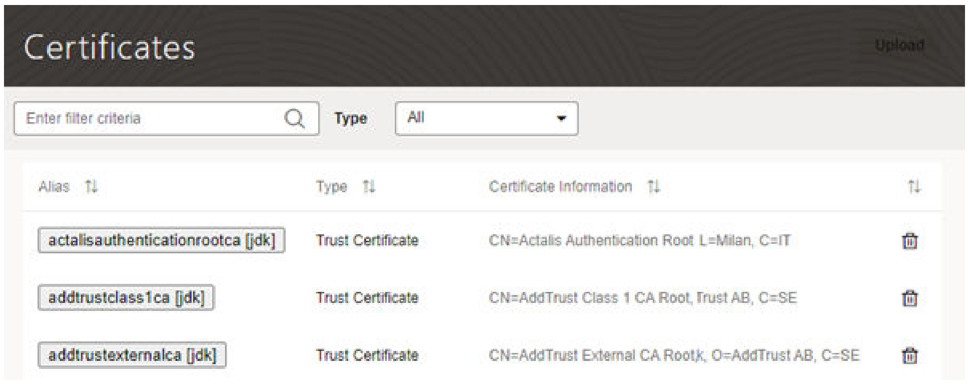
> **Note**
>
> Your staged or published apps might stop working if they use service connections with self-signed certificates and the certificates have expired. Any certificates issued after 2020-09-01T00:00:00.00Z will automatically expire 398 days after they have been issued. If your apps use certificates issued before 2020-09-01T00:00:00.00Z, the certificates will not expire, but you should update them with a newer certificate.
>
> To avoid disruptions, you should plan regular updates to refresh the self-signed certificates before they expire (for example, every 6 months). It's not recommended to use self-signed certificates in production apps.

To upload a self-signed certificate:

1. In the upper-left corner of the Visual Builder title bar, click **Navigation Menu** ☰.
2. Click **Certificates** in the navigation menu to open the Certificates page.



The Certificates page displays a list of the certificates already uploaded to the instance.


Certificates

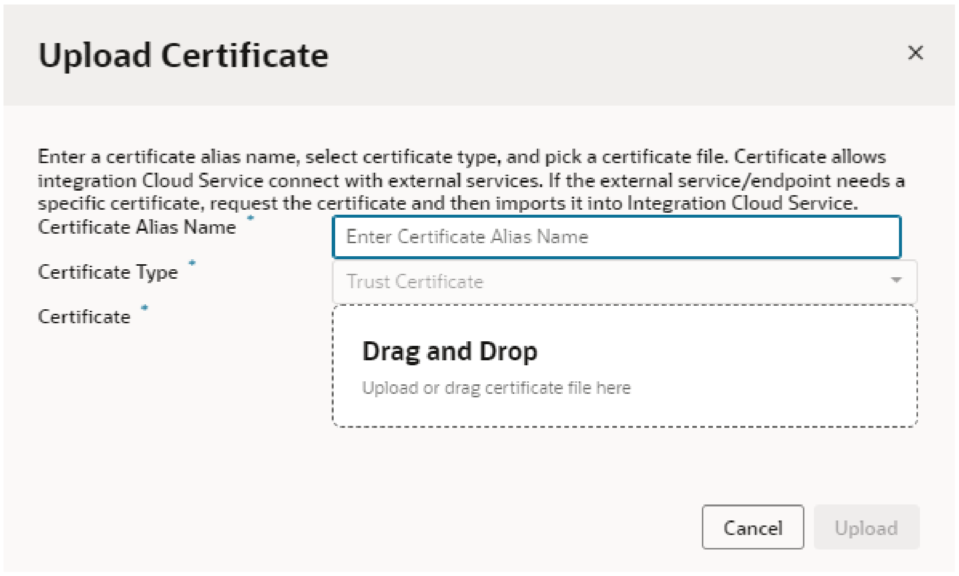
Upload

Enter filter criteria | Type | All

Alias	Type	Certificate Information	
actalisauthenticationrootca [jdk]	Trust Certificate	CN=Actalis Authentication Root L=Milan, C=IT	🗑
addtrustclass1ca [jdk]	Trust Certificate	CN=AddTrust Class 1 CA Root, Trust AB, C=SE	🗑
addtrustexternalca [jdk]	Trust Certificate	CN=AddTrust External CA Rootk, O=AddTrust AB, C=SE	🗑

3. Click **Upload** to open the Upload Certificate dialog box.

   You use the Upload Certificate dialog box to create an alias for the certificate and upload the service's certificate file from your local system.



Enter a certificate alias name, select certificate type, and pick a certificate file. Certificate allows integration Cloud Service connect with external services. If the external service/endpoint needs a specific certificate, request the certificate and then imports it into Integration Cloud Service.

Certificate Alias Name *	Enter Certificate Alias Name
Certificate Type *	Trust Certificate
Certificate *	Drag and Drop Upload or drag certificate file here

4. Type the alias in the Certificate Alias Name field.

   The alias is used to identify the certificate in the table in the Certificates page. The Certificate Type dropdown list is read-only because only Trust Certificates are supported.

5. Drag the certificate file from your local system into the upload target area, or click the upload target area to browse your local system.

6. Click **Upload** to add the certificate to the service keystore.

4

Complete the Post-Migration Tasks

If you have an app that is already live on an Oracle Visual Builder instance and you have migrated the visual application to VB Studio, there are a few things you need to do before you can direct users to the new version of your app published from VB Studio.

Migrate Your Live Application

Before replacing your old live app with the new version published from VB Studio, you'll need to import the data from the old live app. This involves locking the old live app, exporting the data from the live app and then importing it into your new app. You can import the live data as a step in your build pipeline or by importing the data directly into your production environment. You then launch the pipeline to package and deploy the app to the production environment.

The final step is to retire your old app and direct users to the new one. For users who access a web application, you'll need to communicate the new URL that replaces the URL they used previously to access their web applications. For mobile applications, you'll need to rebuild and republish your mobile apps to the app store(s) where you initially published it or use whatever mechanism you used previously to distribute the mobile app so users can update the mobile app on their device.

Lock the Live Visual Builder Application

If you are migrating an application that is a live application, you can lock the live application to prevent any users from modifying any data in the application while you migrate it to your new instance.

To lock an application:

1. Open the Oracle Visual Builder instance.
2. On the Visual Applications home page, open the Application Options menu for the live application you want to lock and select **Lock**.
3. Click **Lock** in the Confirm Lock Application dialog box.

   On the Visual Applications home page, the status of the app is now Live Locked.

Users are not able to use the app to edit data while it is locked.

Export Application Data From the Live Database

You can export all the data contained in your database as CSV files. The export tool creates one CSV file for each of the custom business objects in your database and packages the files as a ZIP archive.

To export the database data as a CSV file:

1. Open the Oracle Visual Builder instance where your old app now has the Live Locked status.
2. On the Visual Applications home page, open the visual application and open the Business Objects pane in the Navigator.

3. Click the **Options** menu in the Business Objects pane and select **Data Manager**.
4. Select the Live database in the dropdown list.
5. Click **Export All Data** to download a ZIP archive that contains CSV files with the data.

After exporting the data, you can import the zip archive into the migrated app in the Deployments page in VB Studio or by configuring the build job for publishing the app to your Oracle Visual Builder production environment.

Import Data into a Deployed Visual Application

You can import data to your deployed visual application if the application was deployed without any data or to replace existing data.

- If your visual application was deployed to a Visual Builder instance in the same identity domain as your VB Studio instance, you can manually import data from your environment's list of deployments:

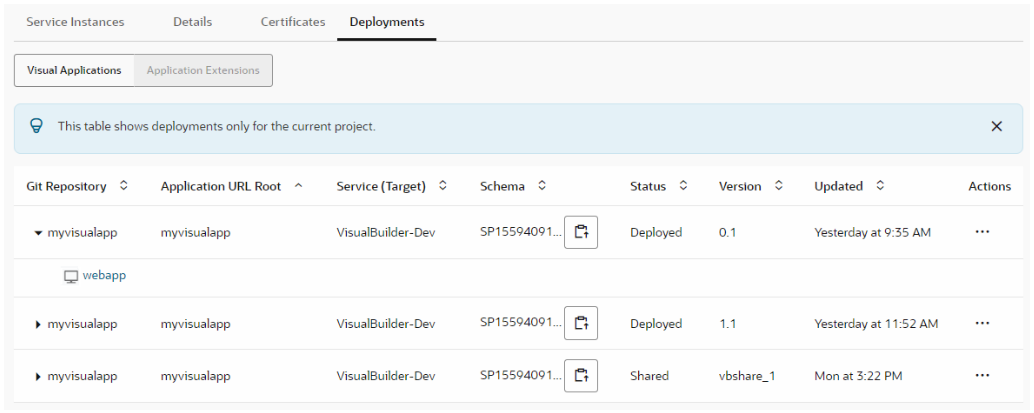
  1. In the VB Studio left navigator, click **Environments**.
  2. Select the environment the visual application is deployed to.
  3. Click **Deployments**, then **Visual Applications**.
  4. For the visual application you want to import data, click **Actions** and select **Import Data**.
  5. In the Import Data dialog, upload a zip archive that contains the data you want to import and click **Import Data**.

     All the data in your environment is deleted and replaced when you import data. When the task is complete, a dialog opens that confirms that the data was successfully imported or warns you if there's a problem.

- If your visual application is deployed to a Visual Builder instance in a different identity domain (say, a production instance) or the instance was added to an environment through credentials, you'll need to add and configure steps in a build job to import data into the application:

  1. In the VB Studio left navigator, click **Builds**.
  2. Click **+ Create Job**. Give the job a new name and a description, select the build executor template, then click **Create**.
  3. On the Job Configuration page, click **Steps**.
  4. Click **Add Step**, select **Visual Application**, then select **Import Data**.
  5. In **Instance**, select the Visual Builder instance where you want to import business object data.
  6. In **Username** and **Password**, enter the user's credentials who can connect to the Visual Builder instance.
  7. In **Application URL Root** and **Application Version**, enter the visual application's root URL and its version. You can find this information from the Deployments tab of the

environment where the visual application is deployed.



This table shows deployments only for the current project.

Git Repository	Application URL Root	Service (Target)	Schema	Status	Version	Updated	Actions
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Deployed	0.1	Yesterday at 9:35 AM	···
webapp							
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Deployed	1.1	Yesterday at 11:52 AM	···
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Shared	vbshare_1	Mon at 3:22 PM	···

8. In **Artifact**, enter the name of a ZIP file that will contain the business object data to import or export; for example, enter `bodata.zip`.
9. Add other steps to the build job to complete the import. For example, you may need to copy an artifact from another job to import business object data. See Copy Artifacts from Another Job in *Using Visual Builder Studio*.
10. Click **Save**.
11. To run the build, click **Build Now**.

Configure a Build Job to Import or Export Data from a Visual Application

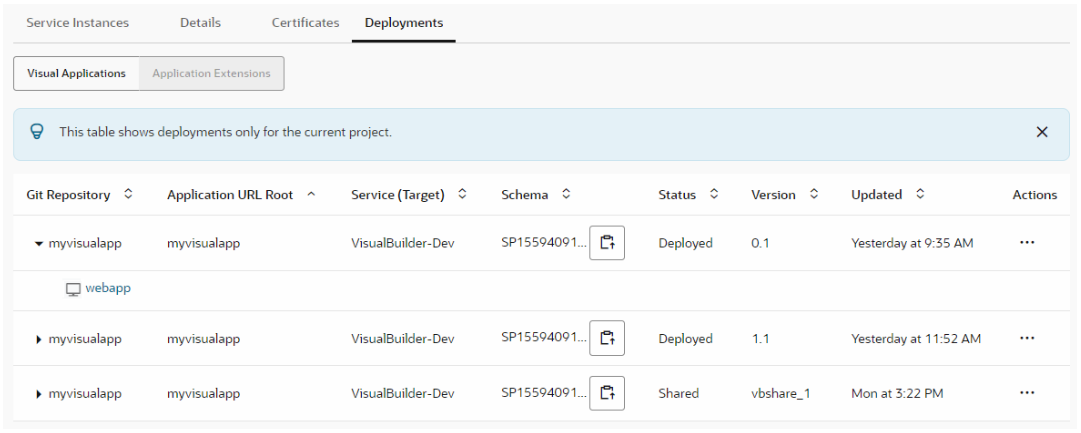
You can import or export data from a visual application through a build job—though a build job is the only way to import or export data when your VB Studio instance and the Visual Builder instance where your visual application is deployed are in different identity domains.

If your instances are in the same identity domain, you can import or export data from your environment's list of deployment using the deployed application's Actions menu.
To import or export data through a build job, you'll need to add the visual application Import Data or Export Data steps to a build job, along with the steps to copy or archive the artifact that contains the business object data. You'll also need the credentials of a user who can access the Visual Builder instance where the visual application is deployed.

1. In the VB Studio left navigator, click **Builds** .
2. In the **Jobs** tab, click **+ Create Job**.
3. In the New Job dialog box, in **Name**, enter a unique name.
4. In **Description**, enter the job's description.
5. In **Template**, select the **System Default OL7 for Visual Builder** template.
6. Click **Create**.
7. Click **Configure** .
8. Click the **Steps** tab.
9. From **Add Step**, select **Visual Application**, and then select **Export Data** or **Import Data**.
10. In **Instance**, select the Visual Builder instance where you want to import or export business object data.

11. In **Username** and **Password**, enter the user's credentials who can connect to the Visual Builder instance.
12. In **Application URL Root** and **Application Version**, enter the visual application's root URL and its version.

    You can find the application's root URL and its version from the **Deployments** tab of the environment where the visual application is deployed.
    Example:



Service Instances | Details | Certificates | Deployments

Visual Applications | Application Extensions

This table shows deployments only for the current project.

Git Repository	Application URL Root	Service (Target)	Schema	Status	Version	Updated	Actions
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Deployed	0.1	Yesterday at 9:35 AM	···
webapp							
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Deployed	1.1	Yesterday at 11:52 AM	···
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Shared	vbshare_1	Mon at 3:22 PM	···

13. In **Artifact**, enter the name of a ZIP file that will contain the business object data to import or export; for example, enter `bodata.zip`.
14. Add other steps to the build job to complete the import or export of the business object data.

    For example, to complete the export of data you'll need to add an after build action that archives the artifact (`bodata.zip`, in our example) while you'll typically need to copy an artifact from another job to import business object data. See Archive Artifacts and Copy Artifacts from Another Job in *Using Visual Builder Studio* for more detail.
15. Click **Save**.
16. To run the build, click **Build Now**.

Run the Pipeline

When you're ready to deploy the visual application to the production instance, run the production pipeline.

1. In the left navigator, click **Builds** .
2. Click the **Pipelines** tab.
3. In development pipeline's row, click the **Actions** menu and select **Run Pipeline**.

After a successful build, you'll find the deployed application's link in the **Deployments** tab of the **Environments** page.

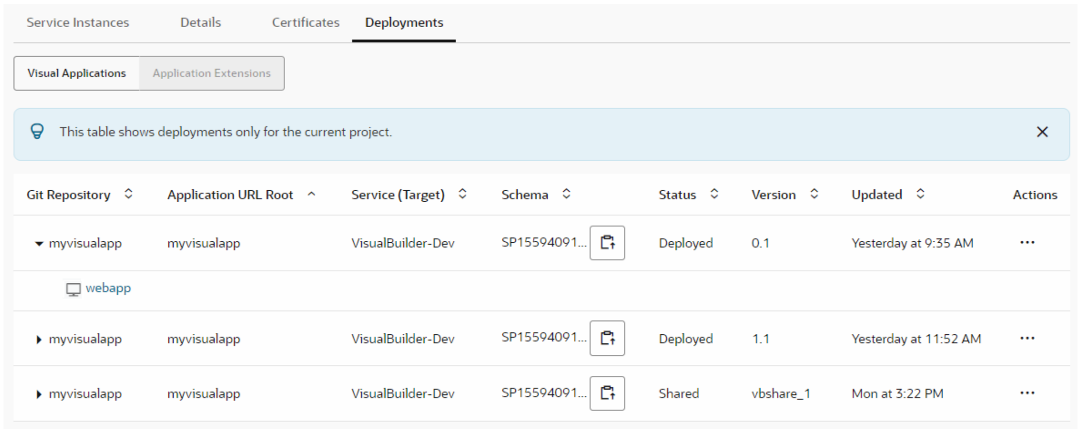
To view the latest build log of a job, open the **Builds** page, click the job's name, and then click **Build Log**.

View the Deployed Visual Application

After the deployment job has successfully run, you can view the deployed applications in the **Deployments** tab of the **Environments** page.

1. In the left navigator, click **Environments** ☁.
2. Select the Visual Builder environment.
3. Click the **Deployments** tab.
4. If not enabled, click the **Visual Applications** toggle button.
5. If the Visual Builder instance is from a different identity domain, provide its access credentials.
6. Expand the app's name to see the deployed app's link.

   The **Deployments** tab displays the applications you've deployed from the current project. It doesn't show applications deployed by other users of the project, or applications deployed from other projects. Here's an example:



This table shows deployments only for the current project.

Git Repository	Application URL Root	Service (Target)	Schema	Status	Version	Updated	Actions
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Deployed	0.1	Yesterday at 9:35 AM	···
webapp							
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Deployed	1.1	Yesterday at 11:52 AM	···
myvisualapp	myvisualapp	VisualBuilder-Dev	SP15594091...	Shared	vbshare_1	Mon at 3:22 PM	···

If you want to undeploy your deployed visual application, you can do so manually or through a job configuration. See Undeploy a Visual Application.

5

Troubleshooting

These topics provide solutions for common issues.

Resolving the Error “Missing rootURL in top level properties of visual-application.json”

If you encounter the error, “Missing rootURL in top level properties of visual-application.json”, after deploying your application to a Visual Builder instance, you’ll need to provide the root URL for the deployed application.

To provide the root URL:

1. Open your application and click the Menu option in the upper-right corner, then select Settings.
2. On the **Application** tab of the Settings editor, enter the root URL in the **Root URL** field.